AD-A043 472   CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF COMPUTER --ETC F/G 9/2
PROBLEMS IN BUILDING AN INSTRUCTABLE PRODUCTION SYSTEM.(U)
JUN 77 M RYCHENER, C FORGY, P LANGLEY   F44620-73-C-0074
UNCLASSIFIED   AFOSR-TR-77-1140   NL

| OF |
AD
A043472

END
DATE
FILMED

9 -77

DDC

AFOSR-TR-77-1140

⑦

Problems in Building an Instructable Production System

M. Rychener, C. Forgy, P. Langley, J. McDermott,
A. Newell, and K. Ramakrishna
June 1977
Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

Interim rept.,

This paper has been submitted for presentation at IJCAI-77, Boston, MA, August, 1977.

2304.   A2

403 081

The Instructable Production System project is exploring the incremental growth properties of production systems (PSs) by constructing a generally intelligent problem-solving system by gradual (external) instruction. The definition of PS and our current architecture are given elsewhere in this volume (Newell, 1977; Forgy and McDermott, 1977). The present task domain is an abstract job shop, in which finished goods are made from raw materials. We start with a Kernel (a small PS of about 200 productions) which has the basic capabilities to grow by instruction: (1) process a restricted natural language; (2) form productions from its input; (3) impose PS control conventions on them; and (4) perform basic manipulations in its environment (Rychener & Newell, 1977). We take the basic computational and representational adequacy of PSs for AI programs as established.

This short note presents some immediate difficulties we expect to encounter. These derive from the instructional situation: (1) The instructor can observe the system in the environment and can communicate with it freely, but cannot examine its internal structure directly. (2) Interaction with the system is in an external language, analogous to natural language. (3) The initiative for interaction is mixed. (4) Instruction may be on any topic: specific tasks, general properties of tasks, the language of communication, possible errors, how to plan and explore, etc. (5) Knowledge and system structure gained through instruction accumulates over the life of the system.

Our current approach uses means-ends analysis as the basic philosophy of both problem-solving and instruction. Goals are symbol structures in Working Memory that describe desired states and processing entirely through the means and tests. Means are encoded as productions that recognize goals and assert subgoals whose satisfaction will achieve the goals. Tests are encoded as productions that recognize the conditions of satisfaction of the goal. The means productions form a means-ends network of goals. Instruction consists of elaborating the nodes of this network as required by a task.

Now for the difficulties on the immediate horizon:

1. Contact: How can initial contact be made with existing knowledge that might be relevant to the task at hand, which is not part of the means-ends network deliberately created by instruction for the task? Use of the data acquired through experience is required in any intelligent agent. Once detected, much processing can be spent on discovering relevance, but initial contact may be extremely difficult. Any general intelligent system will have too much knowledge to consider exhaustively. PS architectures exploit this by storing all knowledge as productions which are evoked only if their conditions "see themselves" in the Working Memory. But means productions are acquired in specific contexts and their conditions become keyed to specific goals and task features. One approach might be to generate variations of current Working Memory goals and data until something is evoked.

2. Incoherency: The PS may be essentially incoherent in describing its situation and difficulties to an external instructor. The means-ends network helps (by providing the same level of explanatory capability as in current expert systems), but is not sufficient.
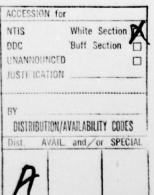
E.g., describe what went wrong from the debris left in Working Memory. The PS's diagnostic and explanatory capabilities are expandable by instruction, but it is currently unclear how this will work.

3. Means-ends analysis efficiency: Initial instruction produces a more elaborate network than is necessary. The instructor uses numerous intermediate goals, both to make his instruction sequences easier to generate and to allow complex procedures to be taught at all; miscommunication leads to a patchwork of variant procedures; the PS uses a goal-encumbered monitoring mode of operation inserting supervisory goals and processes; etc. There are three modes of varying efficiencies: a compiled, efficient mode; ordinary instruction mode; and the monitoring mode. Our primary concern is transforming from the ordinary form to the compiled form, while maintaining a capability to revert back to the other two in debugging situations. This may not be attainable simply through instruction; architectural modification may be required.

4. Utilizing distantly related knowledge: Knowledge about other tasks is imperfect for a given task; it is also embedded in methods and encoded in representations created for (and local to) the distant task. All these aspects cause difficulty, even if contact is made (per 1). A clear symptom will be repetitive instruction to cover minor task variations. One approach is to avoid the difficulty by adopting uniform conventions for encoding. We think this won't work. We favor attempting to map methods to methods (and representations to representations), using ideas from Merlin (Moore & Newell, 1973).

References

Forgy, C. and McDermott, J., 1977. "OPS: A domain-independent production system language", in *Proc. IJCAI-77*.

Newell, A., 1977. "Notes on knowledge representation aspects of production systems", in *Proc. IJCAI-77*.

Moore, J. and Newell, A., 1973. "How can MERLIN understand?", in Gregg, L., Ed., Knowledge and Cognition, Potomac, MD: Lawrence Erlbaum Associates, Pp. 201-252.

Rychener, M. D. and Newell, A., 1977. "An instructable production system: Basic design issues", in D. A. Waterman and F. Hayes-Roth, Eds., *Pattern-Directed Inference Systems*, New York, NY: Academic Press. Forthcoming.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
AFOSR-TR- 77- 1140		

4. TITLE (and Subtitle)

PROBLEMS IN BUILDING AN INSTRUCTABLE
PRODUCTION SYSTEM

5. TYPE OF REPORT & PERIOD COVERED

Interim

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

M. Rychener, C. Forgy, P. Langley,
J. McDermott, A. Newell and K. Ramakrishna

8. CONTRACT OR GRANT NUMBER(s)

F44620-73-C-0074

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Carnegie-Mellon University
Computer Science Dept.
Pittsburgh, PA   15213

10. PROGRAM ELEMENT, PROJECT, TASK
AREA & WORK UNIT NUMBERS

61102F
2304/A2

11. CONTROLLING OFFICE NAME AND ADDRESS
Defense Advanced Research Projects Agency
1400 Wilson Blvd

12. REPORT DATE

June 1977

13. NUMBER OF PAGES

3

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

Air Force Office of Scientific Research (NM)
Bolling AFB, DC   20332

15. SECURITY CLASS. (of this report)

UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING
SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The Instructable Production System project is exploring the incremental growth properties of production systems (PSs) by constructing a generally intelligent problem-solving system by gradual (external) instruction. The definition of PS and our current architecture are given elsewhere in this volume (Newell, 1977; Forgy and McDermott, 1977). The present task domain is an abstract job shop, in which finished goods are made from raw materials. We start with a Kernel (a small PS of about 200 productions) which has the basic capabilities to grow by instruction: (1) process a restricted natural language;

(cont on p 1473 B)

DD FORM 1473   EDITION OF 1 NOV 65 IS OBSOLETE
1 JAN 73
S/N 0102-014-6601

UNCLASSIFIED

20. ABSTRACT (Continued)
(fr P1473A)

(2) form productions from its input; (3) impose PS control conventions on them; and (4) perform basic manipulations in its environment, (Rychener & Newell, 1977). We take the basic computational and representational adequacy of PSs for AI programs as established.)

This short note presents some immediate difficulties we expect to encounter. *that are expected.* These derive from the instructional situation: (1) The instructor can observe the system in the environment and can communicate with it freely, but cannot examine its internal structure directly; (2) Interaction with the system is in an external language, analogous to natural language; (3) The initiative for interaction is mixed; (4) Instruction may be on any topic: specific tasks, general properties of tasks, the language of communication, possible errors, how to plan and explore, etc.; (5) Knowledge and system structure gained through instruction accumulates over the life of the system. *The*

Our current approach uses <u>means-ends analysis</u> as the basic philosophy of both problem-solving and instruction. Goals are symbol structures in Working Memory that describe desired states and processing entirely through the means and tests. <u>Means</u> are encoded as productions that recognize goals and assert subgoals whose satisfaction will achieve the goals. <u>Tests</u> are encoded as productions that recognize the conditions of satisfaction of the goal. The means productions form a means-ends network of goals. Instruction consists of elaborating the nodes of this network as required by a task.

1473B